Reproducible Workflows with IPUMS CPS & the IPUMS Microdata Extract API

February 21, 2023

Questions and Answers

The following are the questions received during the live webinar and their answers. For additional questions or clarifications, contact IPUMS User Support at ipums@umn.edu

Access to the IPUMS API

1. **I already have an IPUMS API key. Do I need to be added as a beta tester to access IPUMS USA and IPUMS CPS data via API?**
   Since the webinar, we are excited to announce that IPUMS USA and IPUMS CPS API functionality have been graduated from beta to stable. You may now access all of our API functionality via the stable API. Instructions are available at developer.ipums.org.

2. **Is it correct that you only need one API key for both IPUMS CPS and IPUMS USA?**
   Your API key will work for all IPUMS data collections that are part of our API program if you are registered to use that specific data collection (e.g., IPUMS USA, IPUMS CPS)

3. **Is there a resource to monitor the development of the ipumsr and ipumspy packages for the IPUMS API?**
   Check in on the GitHub repositories for ipumsr and ipumspy to keep up to date with the latest changes. You can also check out the NEWS file for ipumsr, and the Change Log for ipumspy, which record significant changes corresponding to each release of the respective library.

Workflow

4. **How do you know when to create a function?**

Great question! There's no hard and fast rule; a loose benchmark is if you find yourself copying and modifying the same code 2-3 times or if you would like to be able to apply the same logic to different sets of inputs (for example, different poverty status indicator variables), encapsulating that logic in a function will be beneficial.

5. **Is it possible to over-comment your code?**
   Yes! You should annotate your code to the extent that it helps you or other readers understand the goal of the code and how the lines being annotated are accomplishing that goal. You should not comment every single line of code, but rather use it to clarify places where functionality may not be obvious at first glance. Comments are helpful for reminding yourself (or others) of places where you might be able to do things better when you have more time later or places that you think might cause problems if edge cases are encountered to help guide debugging.

   Also be sure to keep comments up to date as your code changes!

Functionality of the IPUMS API, ipumspy, and ipumsr

6. **What tools do you recommend for working with data from complex surveys in Python? Are there tools for replicate weights? Is there anything comparable to survey/srvyr in R?**
   Unfortunately, as of today, there are no Python libraries that support using replicate weights or incorporating sample design variables.

7. **Is there functionality, current or planned, in ipumsr that's not in ipumspy, or vice versa?**
   We ultimately plan to have parallel functionality in both tools. However, there are differences in the functionality of the two packages at this time.

8. **Does the API extract track the data vintage in case of revisions?**
   The DDI codebook that is used by the ipumspy and ipumsr tools include a citation that has a DOI for the data collection. If you do not include the DDI codebook in your download, you could leverage the date of your API call with a review of the collection-specific DOI table (e.g., here is the DOI version table for CPS) to determine the version/DOI that corresponds to that date. You can also always review the revision history (see CPS revision history for an example) to review changes to the data over time.

9. **What is the frequency of monthly CPS data releases on the IPUMS API? Will monthly data releases be on the same timeline for the API as the IPUMS webpage, or is there a lag?**

CPS data become available via the website and the API at the same time.

10. **What's the best way to programmatically wait for the download of an extract to finish ?**
    In ipumspy, you should use the `wait.for.extract` function. In ipumsr, the equivalent is the `wait_for_extract` function.

11. **Is it possible to subset the data on the read in to save space?**
    The select cases feature from the IPUMS microdata extract system, which allows you to subset data based on certain variables (e.g., the state of Minnesota, women of child bearing age) is not currently supported by the IPUMS API. However, we expect this to become available in the near future.

12. **Is it possible to use the IPUMS API from Stata?**
    Individuals using Stata 16 or higher can consult this blog post on how to make IPUMS extracts from Stata using ipumspy.

13. **If you request a variable available in some survey years but not others?**
    If you request a variable that is only available for some of the samples you have requested, that variable will have missing values in the samples for which it is not available, just as when making an extract through the IPUMS CPS web interface. However, if you request a variable that is not available in *any* of the samples you have included in your extract definition, you will be unable to submit that extract and you will get an error.

14. **The ipumsr API vignette provides a lot of example code for creating and downloading extracts, but what are the essential functions for getting data from the API? More specifically, if I run all the code in the vignette, I end up with three similar-looking objects, called "extract_definition", "downloadable_extract", and "submitted extract" – do I need to create all these objects to get my data?**

    No, you don't need to create all those objects to get your data! The example code creates those objects with descriptive names to highlight what is returned by each function call. `define_extract()` returns an unsubmitted extract definition, `submit_extract()` returns the definition of the submitted extract with the newly-assigned extract number, and `wait_for_extract()` returns the definition of the extract with all the download links attached.

    For an alternative approach, the last section of the vignette provides an example of a workflow for submitting and downloading an extract that does not create any intermediate objects.

IPUMS CPS

15. **Does this example deal with missing values in the SPM and OPM variables?**
    In the IPUMS CPS data, there shouldn't be missing values for SPMPOV and OFFPOV; these are binary (yes/no) values and are available for all persons. In the case of missing values in other variables, it is important to note that IPUMS will classify variable-specific missing values (e.g., NIU/not in universe, “don’t know” responses) with a numeric code. In these cases, you will need to specify how to handle these codes if you do not want them included in your calculations. For “true” missing values (e.g., variables that are not available in certain years), stats packages will ignore these by default.

16. **How are sample ID names created? I’ve noticed depending on the year the same month can end with either an “s” or a “b”?**
    A sample with an s suffix indicates that there is a supplement available for that sample (e.g., Food Security is fielded in December). Samples with b in the suffix do not have a supplement available. A list of all available IPUMS CPS sample ids can be found on this page.